

11/21/00
JC853 U.S. PTO

11-22-00

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

JC914 U.S. PTO
09/11/588
11/21/00

In re Patent Application of: Steven J. Kruij et al.
Title: SEARCHING WITH ADAPTIVELY CONFIGURABLE USER INTERFACE AND EXTENSIBLE
QUERY LANGUAGE
Attorney Docket No.: 777.339US1

PATENT APPLICATION TRANSMITTAL

BOX PATENT APPLICATION

Commissioner for Patents
Washington, D.C. 20231

We are transmitting herewith the following attached items and information (as indicated with an "X"):

- X Return postcard.
X Utility Patent Application under 37 CFR § 1.53(b) comprising:
X Specification (21 pgs, including claims numbered 1 through 32 and a 1 page Abstract).
X Formal Drawing(s) (7 sheets).
X Signed Combined Declaration and Power of Attorney (4 pgs).
X Check in the amount of \$1,806.00 to pay the filing fee.
X Assignment of the invention to Microsoft Corporation (3 pgs) and Recordation Form Cover Sheet.
X Check in the amount of \$40.00 to pay the Assignment recording fee.

The filing fee has been calculated below as follows:

| | No. Filed | No. Extra | Rate | Fee |
|---|-----------|-----------|--------|------------|
| TOTAL CLAIMS | 32 - 20 = | 12 | x 18 = | \$216.00 |
| INDEPENDENT CLAIMS | 14 - 3 = | 11 | x 80 = | \$880.00 |
| [] MULTIPLE DEPENDENT CLAIMS PRESENTED | | | | \$0.00 |
| BASIC FEE | | | | \$710.00 |
| TOTAL | | | | \$1,806.00 |

Please charge any additional required fees or credit overpayment to Deposit Account No. 19-0743.

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. Box 2938, Minneapolis, MN 55402 (612-373-6900)

By: Owen J. Ganton
Atty: Owen J. Ganton
Reg. No. 36,143

Customer Number **21186**

"Express Mail" mailing label number: EL709305397US

Date of Deposit: November 21, 2000

This paper or fee is being deposited on the date indicated above with the United States Postal Service pursuant to 37 CFR 1.10, and is addressed to the Commissioner for Patents, Box Patent Application, Washington, D.C. 20231.

SEARCHING WITH ADAPTIVELY CONFIGURABLE USER INTERFACE AND EXTENSIBLE QUERY LANGUAGE

Copyright Notice and Permission

A portion of this patent document contains material subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent files or records, but otherwise reserves all copyright whatsoever. The following notice applies to this document: Copyright © 1999, Microsoft, Inc.

Technical Field

The present invention concerns data stores, graphical user interfaces, and query languages, particularly, interfaces and query languages.

Background

Data stores, or more particularly databases, generally store files of various forms or structures. Search programs, sometimes called search engines, allow users to use a query language to identify and retrieve these files or specific portions of files based on content-based search criteria.

Although search engines and query languages have conventionally been focused on text files, recent years have seen an increasing need to search other file types, particularly in software development contexts. Seminal examples of file types in the development context include bugs, links, and components. A bug is an abstract data element that refers to a problem report, malfunction, defect, or other related issue regarding software code. A link is a data element identifying a logical association between two or more other data elements or objects, for example two or more duplicate or dependent bugs. Component-type files generally include re-useable software objects, including, for example, binary code, HTML code, source code, and so forth.

Each of these file types often requires a specialized search engine, because they are disparate in form, content, and attributes, they are often stored in different databases, and users have different needs when searching these files than typical text files. Moreover, many of these search engines have a unique graphical user interface tailored to search its associated file types. Thus, for example, a user interface for searching bug files generally includes an arrangement of

input fields, which is entirely different than those in an interface for searching component files. Because of their disparate input-field arrangements, these unique user interfaces force users to continually reorient themselves as they switch from interface to interface to initiate searches for desired files of different types.

5 Some interfaces, particularly OLE DB (Object Linking and Embedding Database) interfaces, provide a measure of commonness or congruence across distinct databases. However, these are limited to defining keyword searches, leaving the bulk of the disparate-interface problem unsolved.

10 Additionally, many of these user interfaces use different query structures, with each query structure optimized for a particular its associated file type. To address this problem, many databases and associated search engines have begun “speaking” a common query language, known as SQL, which reduces the need for users to understand multiple query languages and formats. Although SQL (frequently read and pronounced as “se-kwel”) has a broad range of query properties for searching many file types, its universe of properties and types cannot be
15 readily expanded. Thus, any given version of SQL cannot adapt to search new file types that would require or benefit from new query-language features or properties.

20 Accordingly, the present inventors identified an unmet need for adaptively configurable user interfaces for searching different types of data stores and an unmet need for a readily extensible or expandable query language.

Summary

25 To address these and other problems, the inventors developed an adaptively configurable user interface to facilitate a common user experience across two or more data classes and an extensible common query structure to allow expansion of a query language to meet the demands of new data classes. An exemplary embodiment of the adaptively configurable interface includes a constant portion with one or more input fields (for example, team name, work area, data type, and folder) and an adaptive portion which changes to meet query properties communicated by particular search providers. Once a query is input, the adaptive interface translates it into a common parse tree, which it passes to one or more of the search providers. Which, in turn, map
30 the parse tree onto their own particular query structures before processing. (Some embodiments

pass the query in text form to the search providers.)

An exemplary embodiment of the extensible query structure includes a discovery mechanism for determining whether a database supports or requires use of non-standard query operators and value types. Thus, rather than making assumptions about the query language of a search provider, the exemplary embodiment communicates with the search provider to determine what language-features it supports, allowing the search provider to extend any existing query language or to provide an entirely unique query language for its particular data or file types.

Other notable functionality includes transfer (via email, for example) of queries or partial queries to other users, and storage of queries or partial queries in a team repository for reuse.

Drawings

Figure 1 is a block diagram of an exemplary environment for the invention.

Figure 2 is a block diagram of an exemplary search system or architecture 200 in accord with the invention.

Figure 3 is a flowchart of an exemplary method of operating a search system, such as system 200 in Figure 2, in accord with the invention.

Figure 4 is a facsimile of an exemplary simple query window 400 in accord with the invention.

Figure 5 is a facsimile of an exemplary advanced query window 500 in accord with the invention.

Figure 6A is a facsimile of an exemplary query window 600, which is a variation of the window 500 in Figure 5.

Figure 6B is a facsimile of an exemplary query window 600', which is a variation of window 600 in Figure 6A.

Figure 7 is a facsimile of an exemplary graphical or pictorial query window 700, also in accord with the present invention.

Detailed Description

The following detailed description, which references and incorporates the drawings,

describes and illustrates one or more exemplary embodiments of the invention. These embodiments, offered not to limit but only to exemplify and teach the invention, are shown and described in sufficient detail to enable those skilled in the art to practice the invention. Thus, where appropriate to avoid obscuring the invention, the description may omit certain information known to those of skill in the art.

The description includes three sections. The first describes an exemplary computer system environment for the invention. The second describes exemplary search system architecture, an exemplary method of operation, and exemplary user interface structures, all embodying various aspects of the invention. The third and concluding section summarizes some features and advantages of the exemplary embodiments.

1. Exemplary Environment

Figure 1 is a high-level diagram of an exemplary environment 100 having software 110 and hardware 120 for hosting the invention as executable instructions, data, and/or electronic and mechanical components. However, other suitable environments and variations of the described environment are also possible and within the scope of the invention.

Hardware components 120 are shown as a conventional personal computer (PC) including a number of components coupled together by one or more system buses 121 for carrying instructions, data, and control signals. These buses may assume a number of forms, such as the conventional ISA, PCI, and AGP buses. Some or all of the units coupled to a bus can act as a bus master for initiating transfers to other units. Processing unit 130 may have one or more microprocessors 131 driven by system clock 132 and coupled to one or more buses 121 by controllers 133. Internal memory system 140 supplies instructions and data to processing unit 130. High-speed RAM 141 stores any or all of the elements of software 110. ROM 142 commonly stores basic input/output system (BIOS) software for starting PC 120 and for controlling low-level operations among its components. Bulk storage subsystem 150 stores one or more elements of software 110. Hard disk drive 151 stores software 110 in a nonvolatile form. Drives 152 read and write software on removable media such as magnetic diskette 153 and optical disc 154. Other technologies for bulk storage are also known in the art. Adapters 155 couple the storage devices to system buses 121, and sometimes to each other directly. Other

hardware units and adapters, indicated generally at 160, may perform specialized functions such as data encryption, signal processing, and the like, under the control of the processor or another unit on the buses.

Input/output (I/O) subsystem 170 has a number of specialized adapters 171 for connecting PC 120 to external devices for interfacing with a user. A monitor 172 creates a visual display of graphic data in any of several known forms. Speakers 173 output audio data that may arrive at an adapter 171 as digital wave samples, musical-instrument digital interface (MIDI) streams, or other formats. Keyboard 174 accepts keystrokes from the user. A mouse or other pointing device 175 indicates where a user action is to occur. Block 176 represents other input and/or output devices, such as a small camera or microphone for converting video and audio input signals into digital data. Other input and output devices, such as printers and scanners commonly connect to standardized ports 177. These ports include parallel, serial, SCSI, USB, FireWire, and other conventional forms.

Personal computers frequently connect to other computers in networks. For example, local area network (LAN) 180 connect PC 120 to other PCs 120' and/or to remote servers 181 through a network adapter 182 in PC 120, using a standard protocol such as Ethernet or token-ring. Although Fig. 1 shows a physical cable 183 for interconnecting the LAN, wireless, optical, and other technologies are also available. Other networks, such as wide-area network (WAN) 190 can also interconnect PCs 120 and 120', and even servers 181, to remote computers 191. Computers 181 and 191 have processors, storage, and communications equipment similar to those of PC 120, although usually of higher capacity. Fig. 1 illustrates a communications facility 192 such as a public switched telephone network for a WAN 190 such as an intranet or the Internet. PC 120 can employ an internal or external modem 193 coupled to serial port 177. Other technologies such as packet-switching ISDN, ATM, DSL, frame-relay are also available. In a networked or distributed-computing environment, some of the software 110 may be stored on the other peer PCs 120', or on computers 181 and 191, each of which has its own storage devices and media.

Software elements 110 may be divided into a number of types whose designations overlap to some degree. For example, the previously mentioned BIOS sometimes includes high-level routines or programs which might also be classified as part of an operating system (OS) in

other settings. The major purpose of OS 111 is to provide a software environment for executing application programs 112 and for managing the resources of system 100. An OS such as Windows® or Windows NT® from Microsoft Corp. commonly includes high-level application-program interfaces (APIs), file systems, communications protocols, input/output data conversions, and other functions.

Application programs 112 perform more direct functions for the user. A user normally calls them explicitly, although they can execute implicitly in connection with other applications or by association with particular data files or types. Modules 113 are packages of executable instructions and data, which may perform functions for OSs 111 or for applications 112.

Dynamic link libraries (DLL) and class definitions, for instance, supply functions to one or more programs.

Data 114 includes user data of all types, data generated and/or stored by programs, and digital data that third parties make available on media or by download for use in computer 120. Software elements can be embodied as representations of program instructions and data in a number of physical media, such as memory 140, non-volatile storage 150, and signals on buses 183, 192, and so forth.

2. Exemplary Search System Architecture

Figure 2 shows an exemplary search system or architecture 200 in accord with the present invention. Architecture 200 includes a graphical user interface 202, a search application program interface (API) 204, search providers 204a, 204b, and 204c, and data stores 208a, 208b, and 208c, which in the exemplary embodiment are implemented in C++ or COM. However, other embodiments implement the components as CORBA objects. Indeed, the invention is not limited to any particular form or class of implementations. In some embodiments, the data stores include one or more of the following types of data: files, folders, workspaces (akin to source-code-control (SCC) databases), bugs, problem reports, code defects, project tasks, test cases, test results, build scripts, build results, performance data, load/stress data, and in fact virtually anything related to software under development or the software development process itself.

In general, user interface 202 facilitates user definition of query structures and communication of these query structures to search API 204, which in turn communicates them

one or more of search providers 204a-204c. The search providers process the query structures and search one or more of data stores 208a-208c based on the query structures and return any search results to interface 202 via search API 204.

More particularly, Figure 3 shows an exemplary flow chart 300 that provides further details of operating the exemplary search system in accord with the invention. Flow chart 300 includes blocks 302-314, which are arranged serially in the exemplary embodiment. However, other embodiments of the invention may execute two or more blocks in parallel using two or more independent machines, two or more processors, or a single processor organized as two or more virtual machines, subprocessors, threads, fibers, or other alternative techniques.

Additionally, other embodiments reorder, combine, divide, and/or omit the functions associated with one or more of these exemplary blocks. Moreover, still other embodiments implement the blocks as two or more specific interconnected hardware modules with related control and data signals communicated between and through the modules. Thus, the exemplary process flow is applicable to not only software, but also firmware and hardware implementations.

In process block 302, a user or device invokes a search command, such as a “find command,” indicating a desire to find and retrieve certain files or data within files. In some embodiments, the search command retrieves data in the form of data records, conceptual objects, or other desired logical forms. On invocation of the search command, search API 204 determines the query properties associated with one or more of the search providers to determine one or more query properties, as indicated in block 304.

In some embodiments, this determination entails interrogating one or more of the search providers, each of which responds with one or more sets of data describing one or more of their respective query properties. Other embodiments rely on one or more of the search providers to in effect register one or more of their query properties with API 204 or some other object or resource accessible to API 204 or interface 202. In one variant of this registration, at least one search provider registers one or more its query properties as part of its installation into system 200 and API 204 refers to these registered properties. An exemplary set of query properties includes a list of one or more data property names, query comparison operators, and a number of operands for each of the comparison operators, syntactic data types and data value type for each operand. In some embodiments, the query properties also include localization information about

display strings. After receiving one or more set of query properties from the search providers, execution continues at block 306.

Block 306 entails configuring and displaying (or otherwise manifesting) a user interface based at least in part on the set of query properties for one of the search providers. In the exemplary embodiment, this entails communicating one or more of the query properties for each of the search providers to the user interface.

Based on one or more of these properties, the system configures and displays a graphical user interface having an appropriate set of entry fields, drop-down menus, and/or other conventional mechanisms to facilitate articulation or definition of one or more queries using the particular query properties of one or more of the search providers. (Figures 4, 5, 6A, 6B, and 7, which are discussed below, show several exemplary search windows that are configured based on discovered query properties of the search providers.) After definition of a query, the query is passed from the user interface to search API 204, possibly in a textual form similar to a SQL query or in the form of an XML document, which then executes block 308. In the exemplary embodiment, the search providers communicate a list of one or more searchable fields for its associated data stores. Each field has optionally associated legal values and valid operators, and each operator has associated information indicating the number, style, and type of permissible operands. The searching service also provides APIs to convert between a query tree, a SQL-like representation, XML, and other possible embodiments.

Block 308 entails accepting the query and communicating it to one or more of the search providers. To this end, the exemplary embodiment forms a parse tree representative of the query and passes it by value or by reference to one or more of the search providers. Advantageously, the search provider doesn't need to parse SQL or XML. It has a structurally valid parse tree of the query. This makes query generation much simpler. In many embodiments that pass or call byreference, the search provider is responsible for copying the data and ensuring that the source data remains unmodified. In contrast, the "pass or call by value" approach allows providers to annotate and modify as necessary. This approach is desirable because it allows the provider to modify the parse tree during query formation, optimizing or transforming the tree for the target search facility.

In the exemplary embodiment, search API 204 marshals the parse tree to the search provider. Services are also available to allow the search provider to marshal the parse tree to other search providers, processes, or computers.

Unlike conventional search providers, one or more of the search providers in the exemplary embodiment do not require a parser. In some embodiments, at least one of the search providers includes a parser, which thereby allows sharing of any requisite parsing between search API 204 and the search provider. Some embodiments omit a parser altogether, relying on transfer of the query from API 204 to the search providers in some other form, such as an XML document.

After passing of the query, or more precisely some data structure representative of the query, to one of the search providers, the exemplary method of operation continues at block 310.

In block 310, one or more of the search providers searches its associated data store 208a, 208b, or 208c based on the parse tree or other structure representing the user-defined query. In the exemplary embodiment, one or more of the search providers receiving the parse tree remaps the parse tree according to its query properties to facilitate an optimal search of its associated data stores. For example, a tuple such as CreationDate Between Date1, Date 2, which identifies files (or more generally data objects) having a creation date between Date1 and Date2 inclusive, can be remapped to a combination tuple, such as CreationDate >= Date1 AND CreationDate <= Date 2. Though this example is quite simple, the principle of query transformation allows for remapping arbitrarily complex and abstract operators and queries according to the query grammar of any number of search providers.

Exemplary search providers expose query properties using "script name" and "display name." This allows the search providers to use locale-specific display names, while using locale-independent script names. Specifically, when an exemplary search provider generates a query from a parse tree, it uses the script name, but includes the display name as a column alias. Thus, the query tree points to structures that contain both the display name and the column alias, enabling the search provider to choose the best one for the task at hand. This is also integrated into the SQL-like language and can be integrated into other forms such as XML.

Related functionality in the exemplary embodiment allows for the hierarchical layering or serial chaining of the search providers to answer a query. Specifically, the exemplary

embodiment allows hand off of a query (or equivalent parse tree or XML document) from provider 306c to provider 306b and provider 306b to provider 306a, with the effect of provider 306a fulfilling the query and the results reported back through provider 306c to the user interface via search API 204. This capacity to chain or layer search providers is without theoretical bound.

After completion of the search, the results are returned as indicated in block 312. The exemplary embodiment returns results in the form of OLE DB rowsets. However, other embodiments return results as Active Data Object (ADO) recordsets (a wrapper around OLE DB) or in XML and XQL forms. Other embodiments use other data structures capable of holding multiple search results.

Figures 4, 5, 6A, 6B, and 7 show three exemplary user-selectable types of query windows ---simple, advanced, and graphical--- which can be configured based on discovered query properties of one or more of search providers 206a-206c in Figure 2. Unlike conventional user interfaces that allow searching through two or more search providers, the query windows of the present invention provide users with a consistent experience as they switch from one source provider to another. To achieve consistency across multiple search providers, each query window in the exemplary embodiment includes at least one set of one or more user entry fields which remains constant from one search provider to the next. In the exemplary embodiments, consistency manifest as a set of four fields which have similar placement within the query windows not only for each service provider, but also for each style or type of query window.

In particular, Figure 4 shows an example of a simple query window 400, which includes regions or areas 402, 404, and 406. Region 402 includes input fields 402a, 402b, 402c, and 402d with associated pull-down menus (not shown). (Some embodiments omit the pull-down menus and infer options from context.) In this exemplary embodiment, field 402a accepts entry of one or more team identifiers; field 402b accepts entry of one or more work-area identifiers; field 402c accepts entry of one or more folder identifiers. These identifiers generally designate a set of one or more data stores and/or files to be searched. In the exemplary embodiment, TEAM indicates an input area for identifying one or more servers or data spaces associated with one or more design or development teams. TYPE indicates an input area for identifying the type or types of

objects or data to search; WORK AREA indicates an input area for identifying one or more information collections to search; and FOLDER indicates an input area for further identifying the desired search scope within an identified WORK AREA. Adjacent region 402 is region 404.

The exemplary embodiment reserves region 404 for interface commands. In the exemplary embodiment, one or more of the search providers communicate icons such as icons 404a, 404b, and 404c for association with region 404. Each icon has a meaning determined by the search providers and generally indicates the type of information being search. This embodiment demonstrates searching a single data type (for example, file or bug) per search. However, other embodiments could allow searching across two or more types.

Region 406 includes input fields 406a and 406b, each having an associated pull-down menu. Region 406a accepts entry of a find string, and region 406b accepts entry of one or more look-in identifiers. The look-identifiers generally identify corresponding search spaces within one or more associated databases. Figure 5 shows an example of an exemplary advanced query window 500, which includes a region 502, a region 504, and a region 506. Region 502, like region 402 in Figure 4, includes input fields 502a, 502b, 502c, and 502d with associated pull-down menus (not shown). Region 504 includes a number of interface commands. Region 506 allows users to define or enter one or more name-relationship-value expressions (or tuples) identifying desired data attributes. More precisely, region 506 includes an “and-or” combination column 506a, a name or field column 506b, and one or more value columns 504d. In generally, columns 506b, 506c, and 506d define one or more name-relationship-value tuples. For more than one tuple (or row), column 504a includes, a logical operation (such as “and,” “or,” or “not”) which defines the logical relationship of the tuples. For example, the ordered expression “Product Equals TSIM” in the first row of region 406 identifies data having an associated Product identifier string “TSIM.” The “And’s” in the second and third row of column 506a indicate that the desired data also must satisfy the expressions “Assign_To Equals “KHardy” and Priority Equals “High.”

Figures 6A and 6B show two related variation of window 600. In particular, Figure 6A shows an advanced query window 600 which includes an alias 602 for a tuple or other query structure. Alias 600 is in effect a proxy for a predefined tuple or combination of tuples. Figure 6B shows window 600 after a user had clicked or otherwise selected to display the tuple

underlying alias 602. The exemplary embodiment supports use of one or more aliases in a single query window. (Note that the grouping and hierarchical display are independent of the actual search provider and illustrate how canonical elements can be optimized using shared user-interface techniques.)

5 Relatedly, the exemplary embodiment also facilitates a user identifying and persisting a tuple or combination of tuples as an alias for future use. Moreover, it also allows communicating the alias and its underlying query structure to other users via electronic mail and/or persisting them to a repository for reuse by others. In persisting the query structure, a user has the option to associate with the query a projection list (which identifies the form of the results), ordering
10 information (for example, sort by size, date, etc), search provider, user-interface format parameters (such as, column widths, column ordering, and so forth.

Figure 7 shows a graphical query window 700 that allows a user to define a query pictorially or to view a previously defined query as a graph. Window 700 includes a region 702, a region 704, and a region 706, which are analogous to the regions in the other windows. Region
15 706 includes a structured query diagram noting the conditions (tuples) and groupings. Users can select items or groups of items and create new groupings. Also, new conditions can be added to the diagram using main or context menus. The exemplary embodiment supports in-place editing of the diagram.

One particularly useful feature of user interface 202 and its provision of query windows
20 such as those shown in Figure 4, 5, and 7, is the ability for a user to define all or part of a query in one style or type and to switch from one query window style to another, thereby gaining alternative views of the query structure. For example, a user could begin defining a complex query in an advanced query window, such as window 500 in Figure 5, and then switch, with the click of a mouse, to a graphical query window, such as window 700 in Figure 5, of the same
25 query in a query diagram. Some embodiments indicate to the user when conversion from one query view to another is not logically feasible.

Some embodiments of the invention support side-by-side parallel use of any combination of two of the three query window forms. Thus, as one builds a query in one form of query window, such as the advanced query window, an equivalent query structure manifests in another
30 form of query window, such as the graphical query window, allowing a user to more readily

switch back and forth between the two query windows. Additionally, the exemplary embodiments supports component copying and moving using “cut” and “paste” features. The copying and moving function across different search windows and across different user interface. For example, a set of elements in a graphical query can be selected and pasted into the advanced grid and the result will be a grid representation of the graphical query.

Conclusion

In furtherance of the art, the present inventors have devised an extensible and adaptively configurable user interface to facilitate a common user experience across two or more data classes and an extensible common query structure to allow expansion of a query language to meet the demands of new data classes. The exemplary embodiment includes a discovery mechanism for determining one or more query properties, such as object properties, query operators, and operands, of one or more search providers. The discovery mechanism supports adaptive reconfiguration of a user interface to expose the query properties of the search providers. The exemplary user interface includes first and second regions, with the first region having a set of input fields, which remains substantially constant over a set of two or more search providers, and the second region adaptive to the query properties of a select search provider. Other notable functionality includes transfer (via email, for example) of queries or partial queries to other users, and storage of queries or partial queries in a team repository for reuse.

The embodiments described above are intended only to illustrate and teach one or more ways of practicing or implementing the present invention, not to restrict its breadth or scope. Only the following claims and their equivalents define the actual scope of the invention, which embraces all ways of practicing or implementing the concepts of the invention.

Claims

1. A method of operating a system including two or more search providers and a user interface having a first input region and a second input region, the method comprising:
receiving information regarding one or more query properties from one or more of the search providers;
configuring at least a portion of the second input region of the user interface based on one or more of the received query properties;
receiving user input through the user interface to define a query;
parsing the query to define a parse tree;
passing the parse tree to at least one of the search providers;
transforming the parse tree based on query properties of one of the search providers; and
conducting a search based on the transformed parse tree.
2. The method of claim 1, wherein receiving information regarding one or more query properties comprises receiving information regarding one or data object properties; one or more comparison operators, and one or more operands for each of the one or more comparison operators.
3. The method of claim 1, wherein configuring at least a portion of the second input region of the user interface based on one or more of the received query properties comprises defining and displaying one or more input fields for one or more of the received properties.
4. The method of claim 1, wherein passing the parse tree to at least one of the search providers comprises passing the parse tree by value to one of the search providers.
5. The method of claim 1 further comprising electronically transferring at least a portion of the query to one or more other users.
6. The method of claim 1, further comprising copying at least a portion of the query to a

repository for use by other users.

7. A system comprising:

a graphical user interface having a first input region and a second input region for defining a query;

two or more search providers;

means for receiving information indicating one or more query properties from one or more of the search providers;

means for configuring at least a portion of the second input region of the user interface based on one or more of the received query properties;

means for receiving user input through the user interface to define a query after reconfiguring at least a portion of the second input region;

means for parsing the query to define a first parse tree;

means for passing the parse tree to at least one of the search providers;

means for deriving a second parse tree based on query properties of one of the search providers; and

means for searching a database based on the second parse tree.

8. The system of claim 7, wherein the means for deriving the second parse tree is within one of the two or more search providers.

9. The system of claim 7, wherein the means for conducting a search based on the transformed parse tree is within one of the two or more search providers.

10. A method of operating a system including two or more search providers and a user interface, the method comprising:

receiving information regarding one or more query properties from one or more of the search providers; and

configuring at least a portion of the user interface based on one or more of the received query properties.

11. The method of claim 10, wherein receiving information regarding one or more query properties comprises receiving information regarding one or data object properties; one or more comparison operators, and one or more operands for each of the one or more comparison operators.

12. The method of claim 10, wherein configuring at least a portion of the user interface based on one or more of the received query properties comprises defining and displaying one or more input fields for one or more of the received properties.

13. A method of operating a system including two or more search providers and a user interface, the method comprising:
receiving information regarding one or more query properties from one or more of the search providers; and
communicating a query based on one or more of the received query properties to one or more of the search providers.

14. A method of operating a system including a user interface and one or more search providers, the method comprising:
receiving input through the user interface to define a query;
parsing the query to define a parse tree; and
passing the parse tree to at least one of the search providers.

15. The method of claim 14, wherein passing the parse tree to at least one of the search providers comprises passing the parse tree by value to one of the search providers.

16. A method of operating a system including a user interface and two or more search providers, the method comprising:
passing a parse tree representation of a query to a first one of the search providers;
passing a parse tree representation of the query from the first one of the search providers to a second

one of the search providers; and
transforming the parse tree representation of the query from the first one of search providers, based
on query properties of one of the second one of the search providers.

17. A method comprising:

receiving a signal indicating invocation of a search or find command;

displaying a first part of a graphical user interface in response to receiving the signal;

receiving information regarding one or more query properties from one or more search providers;

and

displaying a second part of the graphical user interface based on the received information
regarding the one or more query properties.

18. A method comprising

obtaining information regarding query definition forms for one or more search providers;

defining a graphical user interface based on the obtained information regarding query
definition forms for at least the one search provider.

19. An extensible search architecture for a computer system, comprising:

means for determining query properties of two or more search providers; and

means for communicating queries based on the determined query properties of the search
providers to one or more of the search providers.

20. An extensible search architecture for a computer system, comprising:

a graphical user interface;

two or more search providers; and

an interface coupled between the two or more search providers and the graphical user interface
and adapted to communicate information regarding query definition forms from one or more of
the search providers to the graphical user interface or information regarding one or more queries

from the graphical user interface to the search providers.

21. A method for providing a unified user interface to two or more search providers, each adapted to search a different data store and each having one or more unique query operators or query-definition fields, the method comprising:

identifying at least the unique query operators and query-definition fields supported by each search provider;

displaying a user interface that allows definition of a query based on one or more of the unique query operators or query-definition fields; and

processing the query by passing information regarding the one or more unique query operators and query-definition fields in the query to at least one of the search providers.

22. The method of claim 21, wherein identifying at least the unique query operators and query-definition fields supported by each search provider comprises receiving information from each search provider identifying its unique query operators and query definition fields.

23. The method of claim 21, wherein displaying a user interface that allows definition of a query based on one or more of the unique query operators or query-definition fields comprises displaying the one or more unique query operators or query-definition fields.

24. The method of claim 21, wherein processing the query by passing information regarding the one or more unique query operators and query-definition fields in the query to at least one of the search providers comprises passing a parse tree based on the query to at least one of the search providers.

25. A computer readable medium having executable instructions encoded thereon comprising:

a plurality of search providers, each adapted to search at least one data store using a supported set of search operators and a supported set of search fields;

a configurable user interface having two or more interface elements which allow construction of a query using any of the set of supported search operators and search fields from any of the plurality of search providers; and

a search application program interface between the configurable user interface and the plurality of search providers, the search application program interface adapted to provide information regarding the supported set of search operators and search fields for each of the search providers so that the user interface can be configured to allow access to search operators and search fields supported by any of the plurality of search providers.

26. The medium of claim 25, wherein one or more portions of the search providers, the user interface, or the search application program interface are implemented in accord with COM..

27. The medium of claim 25, wherein one or more of the search providers are implemented in an object-oriented programming language.

28. The medium of claim 25, wherein one or more of the search providers is adapted to search two or more data stores.

29. The medium of claim 25, wherein the plurality of search providers includes at least three search providers.

30. The medium of claim 25, wherein the plurality of search providers includes a first search provider for searching bug files, a second search provider for searching link files, and a third search provider for searching software component files.

31. . A method of operating a graphical user interface for defining queries, the method comprising:
receiving user input defining at least a portion of a query, through a table-style interface window;
and
displaying a diagrammatic- or keyword-style interface window based on the query.

32. A method of operating a graphical user interface for defining queries, the medium comprising:
displaying a query in a form of a diagram or a table within a first window of the graphical user interface; and
displaying the query in a different form within a second window of the graphical user interface.

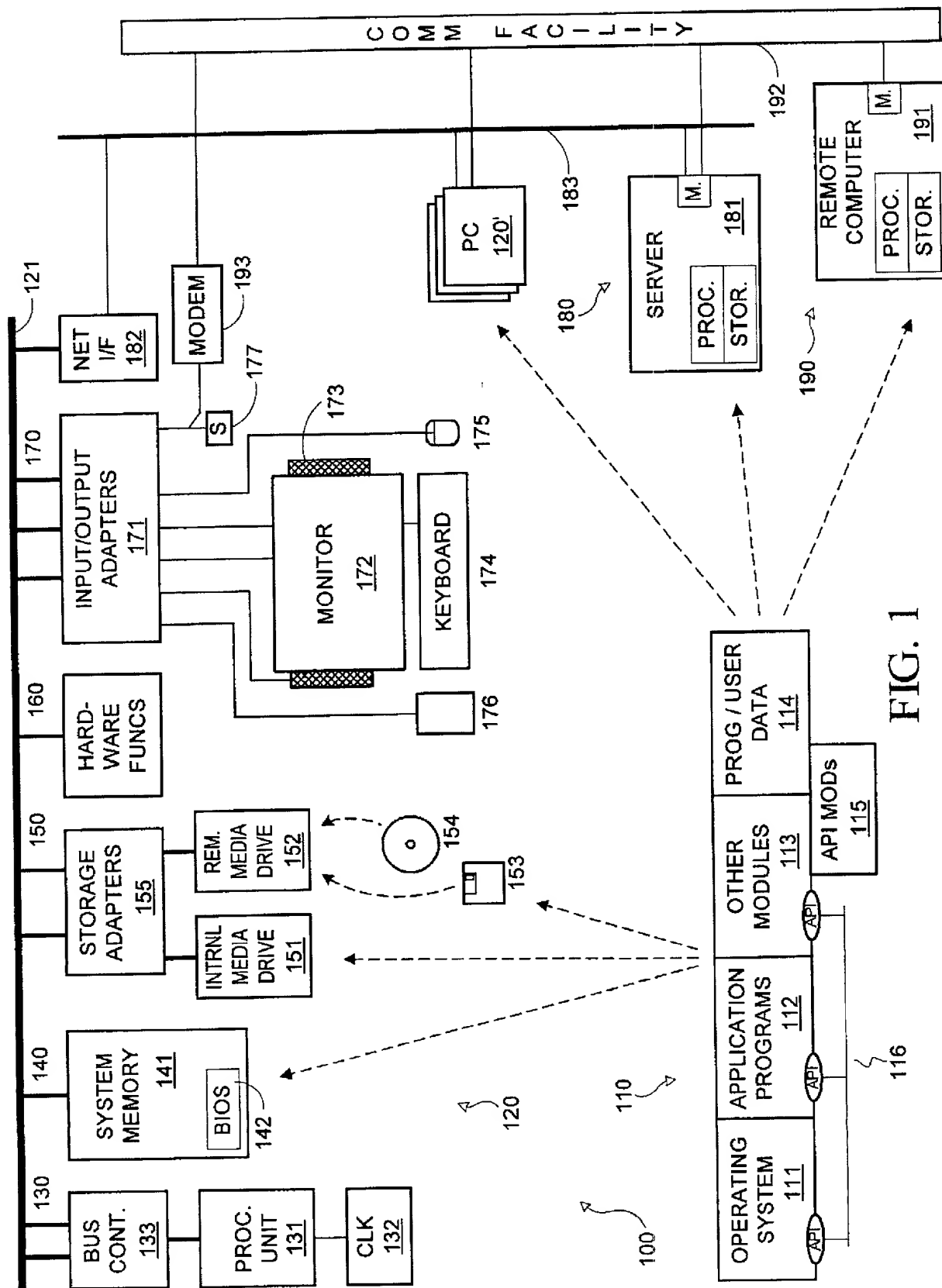
Abstract

Most search engines, include user interfaces and query languages which are tailored for use with specific file types. Thus, a user interface for searching one type file can be quite different than an interface for searching another file type. Disparate user interfaces are troublesome since users must continually reorient themselves as they switch from interface to interface to initiate searches for different file types. Additionally, many of these user interfaces use different query languages and structures, which are not generally adaptive to new file types and query language features. Accordingly, the present inventors developed an adaptively configurable user interface to facilitate a common user experience across two or more databases and an extensible common query structure to allow expansion of a query language to meet the demands of new file types. An exemplary embodiment includes a discovery mechanism for determining query properties of search providers, thereby facilitating adaptive configuring of a user interface to expose the determined query properties. An exemplary embodiment of the user interface includes a portion, which remains relatively constant across two or more search providers to reduce user disorientation.

"Express Mail" mailing label number: EL709305397 US

Date of Deposit: November 21, 2000

This paper or fee is being deposited on the date indicated above with the United States Postal Service pursuant to 37 CFR 1.10, and is addressed to the Commissioner for Patents, Box Patent Application, Washington, D.C. 20231.



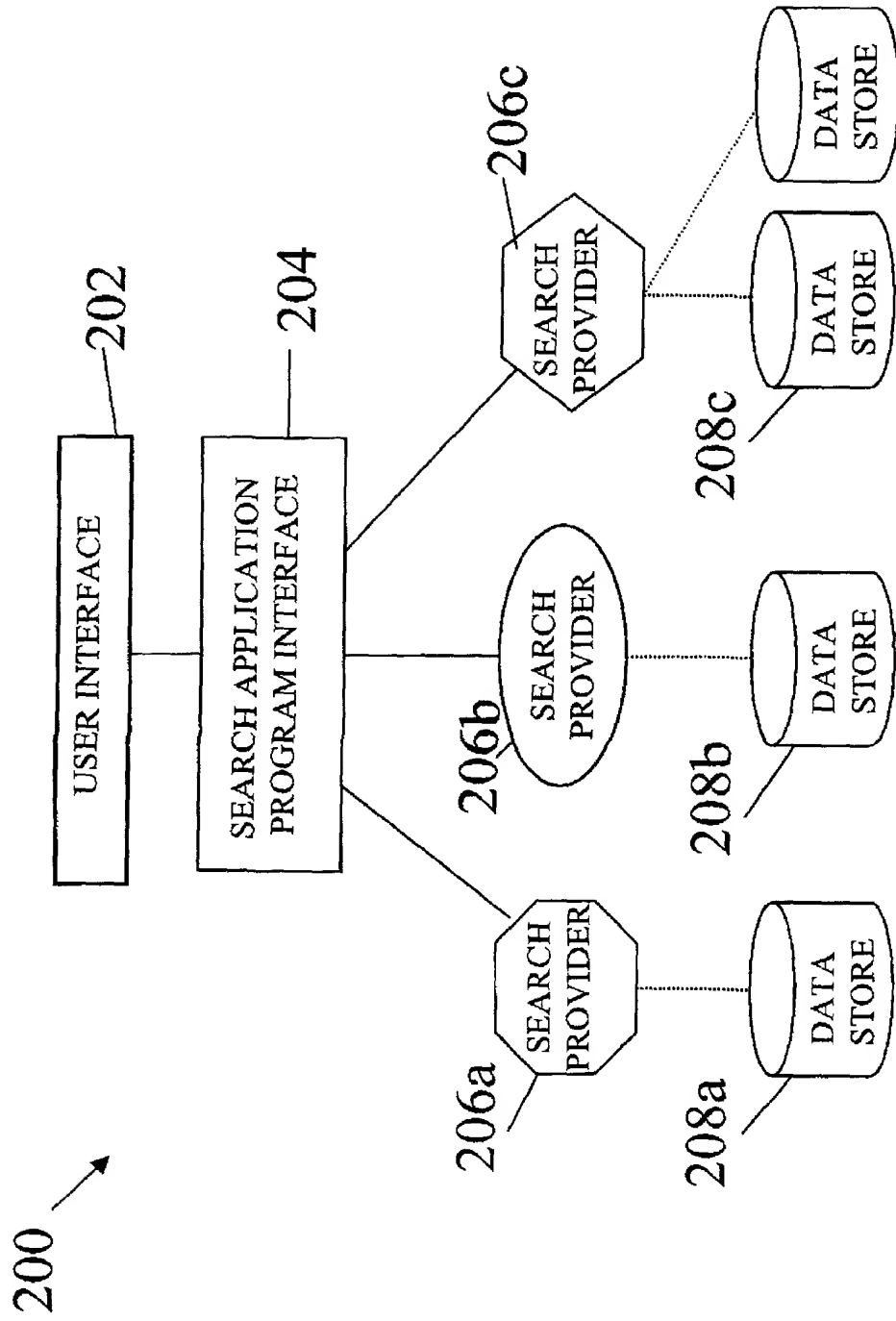


FIG. 2

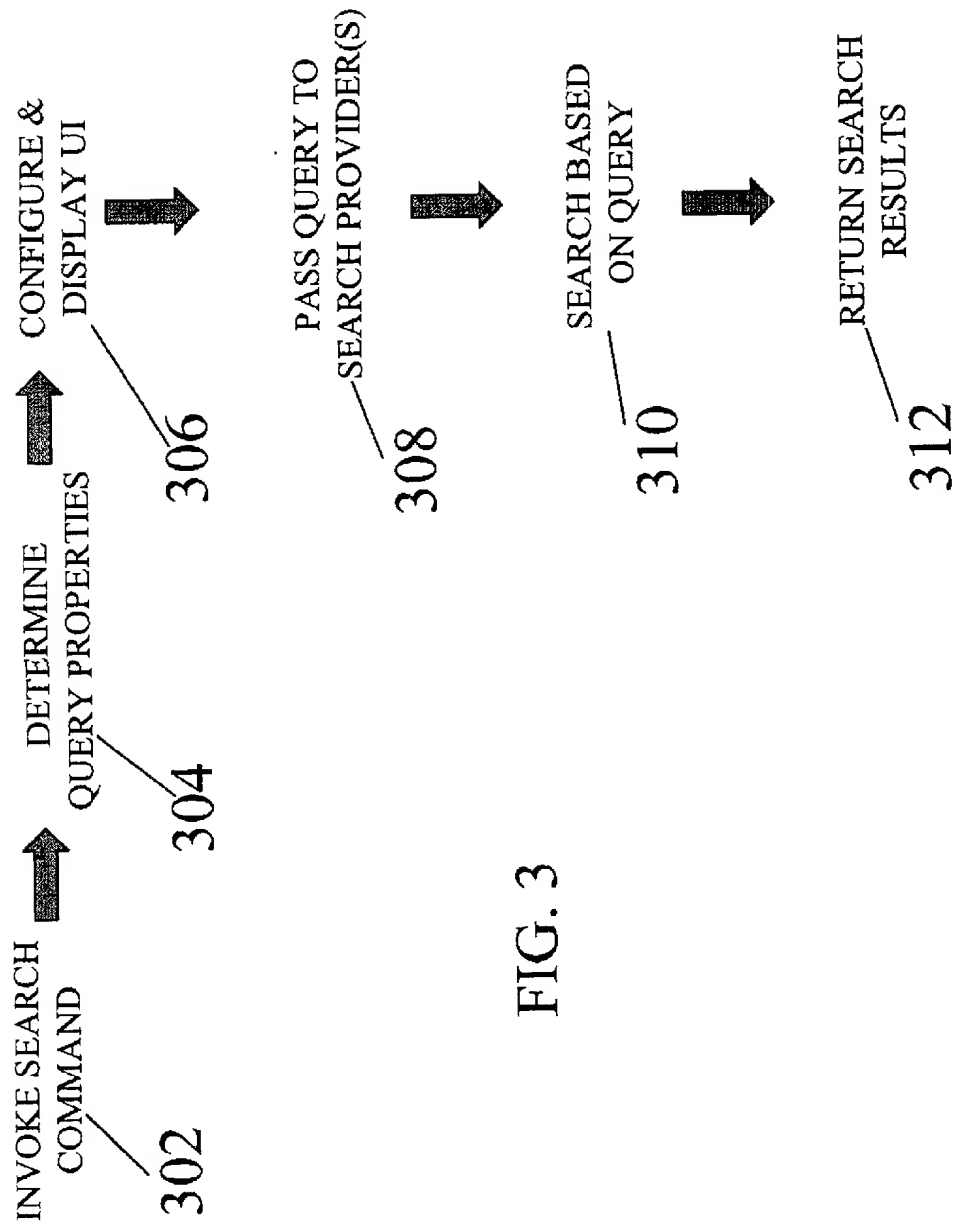


FIG. 3

09/15/2009 10:40:00

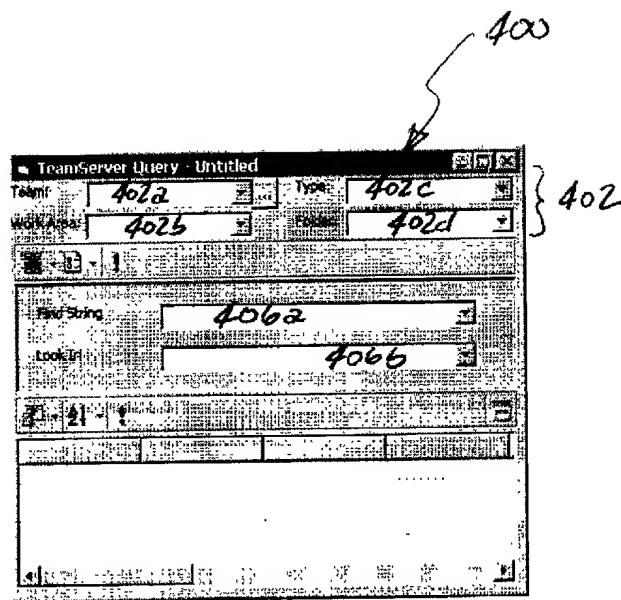


FIG. 4

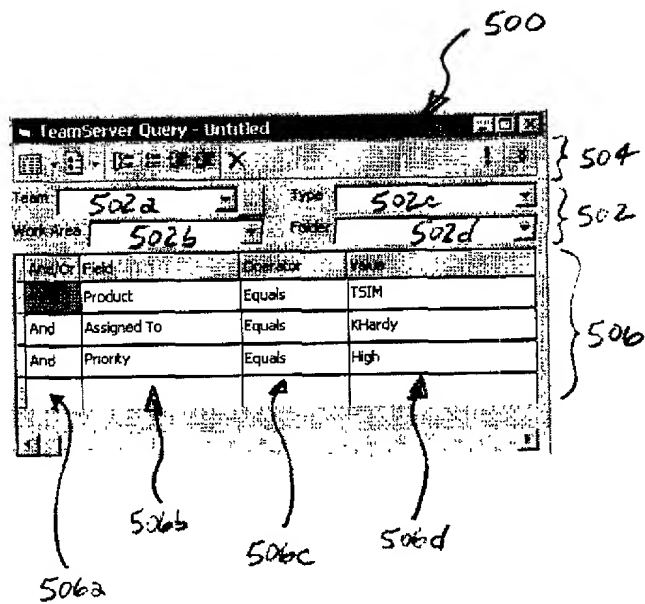


FIG. 5

FIG. 6A

TeamServer Query - Untitled

Team: Type:

Work Area: Folder:

| And/Or | Field | Operator | Value |
|--------|--------------------|----------|-------|
| | Product | Equals | TSIM |
| And | High Priority bugs | | |

602

FIG. 6B

TeamServer Query - Untitled

Team: Type:

Work Area: Folder:

| And/Or | Field | Operator | Value |
|--------|-------------|----------|--------|
| | Product | Equals | TSIM |
| And | Assigned To | Equals | KMarty |
| And | Priority | Equals | High |

602'

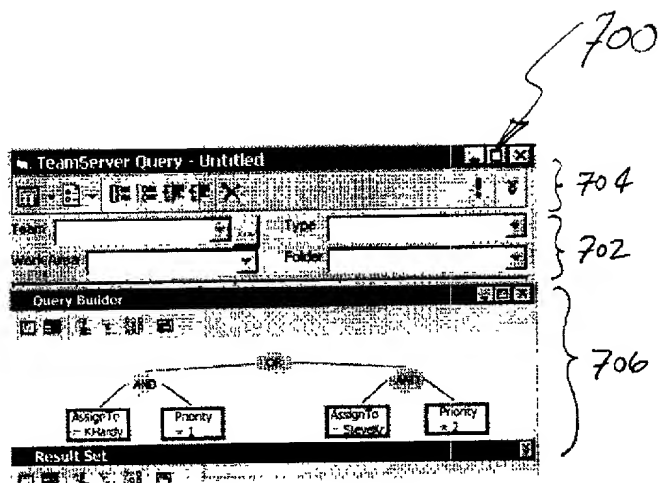


FIG. 7

SCHWEGMAN LUNDBERG WOESSNER KLUTH

United States Patent Application

COMBINED DECLARATION AND POWER OF ATTORNEY

As a below named inventor I hereby declare that: my residence, post office address and citizenship are as stated below next to my name; that

I verily believe I am the original, first and joint inventor of the subject matter which is claimed and for which a patent is sought on the invention entitled: **SEARCHING WITH ADAPTIVELY CONFIGURABLE USER INTERFACE AND EXTENSIBLE QUERY LANGUAGE.**

The specification of which is attached hereto.

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the patentability of this application in accordance with 37 C.F.R. § 1.56 (attached hereto). I also acknowledge my duty to disclose all information known to be material to patentability which became available between a filing date of a prior application and the national or PCT international filing date in the event this is a Continuation-In-Part application in accordance with 37 C.F.R. §1.63(e).

I hereby claim foreign priority benefits under 35 U.S.C. §119(a)-(d) or 365(b) of any foreign application(s) for patent or inventor's certificate, or 365(a) of any PCT international application which designated at least one country other than the United States of America, listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on the basis of which priority is claimed:

No such claim for priority is being made at this time.

I hereby claim the benefit under 35 U.S.C. § 119(e) of any United States provisional application(s) listed below:

No such claim for priority is being made at this time.

I hereby claim the benefit under 35 U.S.C. § 120 or 365(c) of any United States and PCT international application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States or PCT international application in the manner provided by the first paragraph of 35 U.S.C. § 112, I acknowledge the duty to disclose material information as defined in 37 C.F.R. § 1.56(a) which became available between the filing date of the prior application and the national or PCT international filing date of this application:

No such claim for priority is being made at this time.

I hereby appoint the following attorney(s) and/or patent agent(s) to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith:

| | | | | | |
|------------------------|-----------------|-----------------------------|-----------------|-----------------------|-----------------|
| Anglin, J. Michael | Reg. No. 24,916 | Harris, Robert J. | Reg. No. 37,346 | Nelson, Albin J. | Reg. No. 28,650 |
| Beekman, Marvin L. | Reg. No. 38,377 | Huebsch, Joseph C. | Reg. No. 42,673 | Nielsen, Walter W. | Reg. No. 25,539 |
| Bianchi, Timothy E. | Reg. No. 39,610 | Jurkovich, Patti J. | Reg. No. 44,813 | Oh, Allen J. | Reg. No. 42,047 |
| Billion, Richard E. | Reg. No. 32,836 | Kalis, Janal M. | Reg. No. 37,650 | Padys, Danny J. | Reg. No. 35,635 |
| Black, David W. | Reg. No. 42,331 | Kaufmann, John D. | Reg. No. 24,017 | Parker, J. Kevin | Reg. No. 33,024 |
| Brennan, Leoniede M. | Reg. No. 35,832 | Klima-Silberg, Catherine I. | Reg. No. 40,052 | Perdok, Monique M. | Reg. No. 42,989 |
| Brennan, Thomas F. | Reg. No. 35,075 | Kluth, Daniel J. | Reg. No. 32,146 | Prout, William F. | Reg. No. 33,995 |
| Brooks, Edward J., III | Reg. No. 40,925 | Lacy, Rodney L. | Reg. No. 41,136 | Sako, Katie E. | Reg. No. 32,628 |
| Chu, Dinh C.P. | Reg. No. 41,676 | Lemaire, Charles A. | Reg. No. 36,198 | Schumm, Sherry W. | Reg. No. 39,422 |
| Clark, Barbara J. | Reg. No. 38,107 | LeMoine, Dana B. | Reg. No. 40,062 | Schwegman, Micheal L. | Reg. No. 25,816 |
| Clise, Timothy B. | Reg. No. 40,957 | Lundberg, Steven W. | Reg. No. 30,568 | Scott, John C. | Reg. No. 38,613 |
| Crouse, Daniel D. | Reg. No. 32,022 | Maeyaert, Paul L. | Reg. No. 40,076 | Smith, Michael G. | Reg. No. 45,368 |
| Dahl, John M. | Reg. No. 44,639 | Maki, Peter C. | Reg. No. 42,832 | Speier, Gary J. | Reg. No. 45,458 |
| Drake, Eduardo E. | Reg. No. 40,594 | Malen, Peter L. | Reg. No. 44,894 | Steffey, Charles E. | Reg. No. 25,179 |
| Embretson, Janet E. | Reg. No. 39,665 | Mates, Robert E. | Reg. No. 35,271 | Terry, Kathleen R. | Reg. No. 31,884 |
| Fordenbacher, Paul J. | Reg. No. 42,546 | McCrackin, Ann M. | Reg. No. 42,858 | Tong, Viet V. | Reg. No. 45,416 |
| Forrest, Bradley A. | Reg. No. 30,837 | Moore, Charles L., Jr. | Reg. No. 33,742 | Viksnins, Ann S. | Reg. No. 37,748 |
| Gamon, Owen J. | Reg. No. 36,143 | Nama, Kash | Reg. No. 44,255 | Woessner, Warren D. | Reg. No. 30,440 |

I hereby authorize them to act and rely on instructions from and communicate directly with the person/assignee/attorney/firm/organization/who/which first sends/sent this case to them and by whom/which I hereby declare that I have consented after full disclosure to be represented unless/until I instruct Schwegman, Lundberg, Woessner & Kluth, P.A. to the contrary.

Please direct all correspondence in this case to **Schwegman, Lundberg, Woessner & Kluth, P.A.** at the address indicated below:

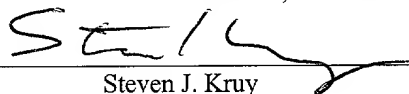
P.O. Box 2938, Minneapolis, MN 55402

Telephone No. (612)373-6900

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of joint inventor number 1 : **Steven J. Kruey**
Citizenship: **United States of America**
Post Office Address: **6330 286th Place NE
Carnation, WA 98014**


Residence: **Carnation, WA**

Signature: 
Steven J. Kruey

Date: 11/16/2000

Full Name of joint inventor number 2 : **Christopher G. Kaler**
Citizenship: **United States of America**
Post Office Address: **22222 NE 28th Place
Redmond, WA 98053**

Residence: **Redmond, WA**

Signature: 
Christopher G. Kaler

Date: 11/16/2000

☒ Additional inventors are being named on separately numbered sheets, attached hereto.

Full Name of joint inventor number 3 : **Kenneth D. Hardy**

Citizenship: **United States of America**

Residence: **Redmond, WA**

Post Office Address: 4306 156th Avenue NE
Redmond, WA 98052

Signature: _____

Date: _____

Kenneth D. Hardy

Full Name of joint inventor number 4 : **Jomo Ahab Fisher**

Citizenship: **United States of America**

Residence: **Redmond, WA**

Post Office Address: 16646 NE 39th Way
#P2047
Redmond, WA 98052

Signature: _____

Date: _____

Jomo Ahab Fisher

Full Name of inventor:

Citizenship:

Residence:

Post Office Address:

Signature: _____

Date: _____

Full Name of inventor:

Citizenship:

Residence:

Post Office Address:

Signature: _____

Date: _____

§ 1.56 Duty to disclose information material to patentability.

(a) A patent by its very nature is affected with a public interest. The public interest is best served, and the most effective patent examination occurs when, at the time an application is being examined, the Office is aware of and evaluates the teachings of all information material to patentability. Each individual associated with the filing and prosecution of a patent application has a duty of candor and good faith in dealing with the Office, which includes a duty to disclose to the Office all information known to that individual to be material to patentability as defined in this section. The duty to disclose information exists with respect to each pending claim until the claim is canceled or withdrawn from consideration, or the application becomes abandoned. Information material to the patentability of a claim that is canceled or withdrawn from consideration need not be submitted if the information is not material to the patentability of any claim remaining under consideration in the application. There is no duty to submit information which is not material to the patentability of any existing claim. The duty to disclose all information known to be material to patentability is deemed to be satisfied if all information known to be material to patentability of any claim issued in a patent was cited by the Office or submitted to the Office in the manner prescribed by §§ 1.97(b)-(d) and 1.98. However, no patent will be granted on an application in connection with which fraud on the Office was practiced or attempted or the duty of disclosure was violated through bad faith or intentional misconduct. The Office encourages applicants to carefully examine:

- (1) prior art cited in search reports of a foreign patent office in a counterpart application, and
- (2) the closest information over which individuals associated with the filing or prosecution of a patent application believe any pending claim patentably defines, to make sure that any material information contained therein is disclosed to the Office.

(b) Under this section, information is material to patentability when it is not cumulative to information already of record or being made of record in the application, and

- (1) It establishes, by itself or in combination with other information, a prima facie case of unpatentability of a claim; or
- (2) It refutes, or is inconsistent with, a position the applicant takes in:
 - (i) Opposing an argument of unpatentability relied on by the Office, or
 - (ii) Asserting an argument of patentability.

A prima facie case of unpatentability is established when the information compels a conclusion that a claim is unpatentable under the preponderance of evidence, burden-of-proof standard, giving each term in the claim its broadest reasonable construction consistent with the specification, and before any consideration is given to evidence which may be submitted in an attempt to establish a contrary conclusion of patentability.

(c) Individuals associated with the filing or prosecution of a patent application within the meaning of this section are:

- (1) Each inventor named in the application;
- (2) Each attorney or agent who prepares or prosecutes the application; and
- (3) Every other person who is substantively involved in the preparation or prosecution of the application and who is associated with the inventor, with the assignee or with anyone to whom there is an obligation to assign the application.

(d) Individuals other than the attorney, agent or inventor may comply with this section by disclosing information to the attorney, agent, or inventor.